CHAPTER

11

PERFORMANCE TUNING

After reading this chapter and completing the exercises, you will be able to:

- Understand the performance and monitoring tools of Windows 2000
- Create a Counter log for historical analysis
- Create Alert events to warn of performance problems
- Create a baseline
- Detect and eliminate bottlenecks

Once you have installed and configured Windows 2000 Professional, connected it to the network, and set up printers, you are ready to optimize your computer performance. Windows 2000 includes several tools for monitoring your computer's performance and tuning it for the best output. Tools discussed include the Performance Console, Event Viewer, and Task Manager.

We introduce and describe these tools and discuss specific system objects and counters that are worth monitoring. You learn what combinations of counters can be used to analyze system slowdowns and how to isolate, identify, and correct system bottlenecks. Very few operating systems include the kinds of tools that Windows 2000 offers to help inspect and analyze system performance. In this chapter, you learn how to use these Windows 2000 monitoring tools to good effect.

Establishing a Baseline

To recognize when bottlenecks exist, it's first necessary to establish some feeling for what's normal on your system, a **baseline** against which you can measure system behavior. Key elements in a baseline include recorded observations about the characteristics and behavior of the computer system. Baselines can be formed by creating a Counter log for the list of counters you consider important and collecting data for those counters over a period of time at regular intervals. This helps you establish a definition of what a normal load looks like and provides points of comparison for future system behavior.

Of course, you'll want to make sure that your system baseline doesn't itself indicate existing bottlenecks. If you discover unacceptably long queues or evidence of memory problems when you create a baseline, you'll want to address whatever bottlenecks you discover right away. We discuss how you can do this for common Windows 2000 subsystems in the sections that follow.

Monitoring and Performance Tuning

When it comes to system analysis, there are two primary components involved in tackling performance-related issues:

- *Monitoring:* This requires a thorough understanding of system components and their behavior, as well as observation of those components and how they behave on a regular basis.
- *Performance tuning:* This activity consists of changing a system's configuration systematically, and carefully observing performance before and after such changes. Changes that improve performance should be left in place; those that make no difference—or that make things worse—should be reversed. There are many ways to improve Windows 2000 system performance. The more useful of these approaches or configuration changes are covered in this chapter.

In many ways, Windows 2000 does a remarkable job of tuning itself. It is capable of managing both its physical and virtual memory quite well. It also adjusts how it allocates memory dynamically and effectively among a variety of uses, including file caching, virtual memory, system kernel use, and applications use. Because of all of these self-tuning features, Windows 2000 offers a more limited set of tools and utilities to monitor and alter system performance than do older operating systems, such as Windows NT. Changing the Windows 2000 operating system configuration is rarely required. Instead, you learn how to recognize and react to system bottlenecks that can limit a system's overall performance.

Task Manager

Windows Task Manager, shown in Figure 11-1, provides an overview of the current state of a computer. You can access the Task Manager in one of three ways:

- Press Ctrl+Alt+Delete and click the Task Manager button on the Windows Security window.
- Press Ctrl+Shift+Esc.
- Right-click any unoccupied area on the Windows 2000 taskbar and select Task Manager from the menu that appears.

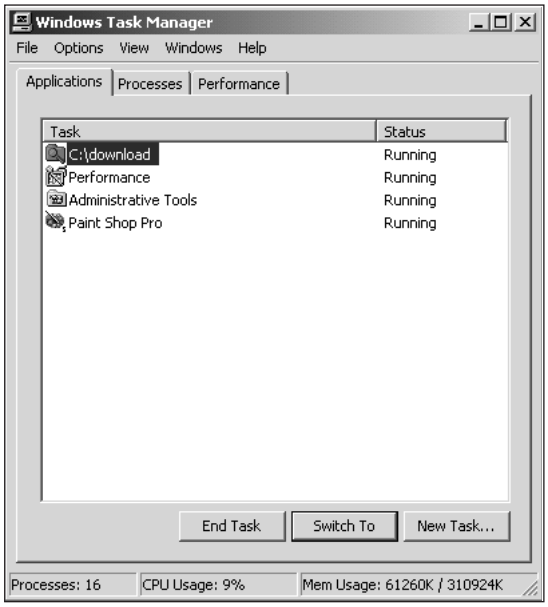

Figure 11-1 Task Manager, Applications tab

The first tab that appears in Task Manager is the Applications tab, which is shown in Figure 11-1. This tab displays all programs currently running on the computer and the status of those programs (usually "Running"). You can use this tab to halt an application by highlighting an entry in the list and clicking the End Task button. To switch to a specific task, highlight an entry and click the Switch To button. To launch a new application, click the New Task button and provide the name of an executable program or command in the Create New Task dialog box that appears.

The Processes tab shows all currently active processes with information about each, including its Process ID number (PID), CPU usage (CPU), CPU time, and Memory Usage. A **process** is an environment that defines the resources available to threads, the executable parts of an application. This display is an excellent instant diagnostic tool to show when ill-behaved applications take up an inordinate amount of CPU time. If this happens at the moment you

use Task Manager, you'll see the process's CPU usage spike above 90%. Even if an application is not currently hogging the CPU, the CPU time entry might be high enough (above 80%) to stick out like a sore thumb.

You can change the columns displayed on the Processes tab by choosing Select Columns from the View menu. This tab lists all processes that contribute to the operation of Windows 2000, including Winlogon.exe and Lsass.exe. You can stop any process by selecting it from the list, then clicking the End Process button.

Be wary of ending Windows 2000 processes; you can cripple or disable a system by ending processes that are required for system operation.

The Performance tab, shown in Figure 11-2, provides a graphical representation of cumulative CPU usage and memory usage. The four text windows at the bottom of the screen provide detailed information on the total number of handles, threads, and processes (Totals) active on the system, the amount of memory allocated to application programs or the system (Commit Charge), the amount of physical memory installed on your computer (Physical Memory), and the memory used by the operating system for internal processes (Kernel Memory). A **thread** is the minimal unit of system execution and corresponds roughly to a task within an application, within the Windows 2000 kernel, or within some other major system component, whereas a **handle** is an internal identifier for some kind of system resource, object, or other component that must be accessed by name.

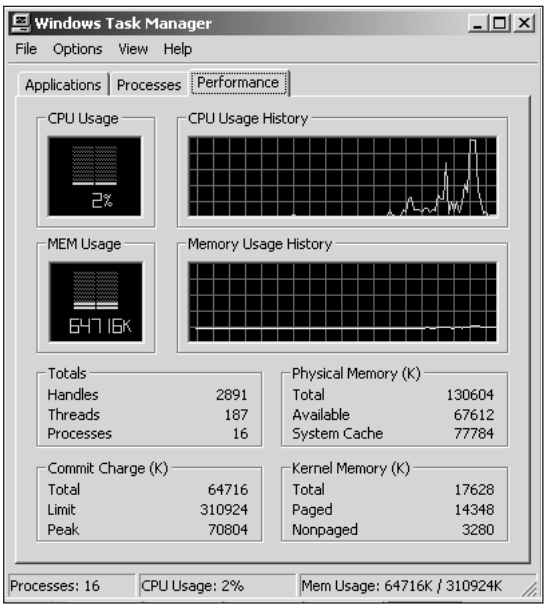

Figure 11-2 Task Manager, Performance tab

You can use the Performance tab in Task Manager to quickly ascertain whether a computer is performing optimally. If the total CPU usage shown in the status bar is consistently high—say over 70%—you can use the Processes tab to identify the process that is monopolizing the CPU, and take corrective action.

System Monitor

The performance monitoring tool included with Windows 2000 can monitor and track many different areas of system performance. Called **System Monitor**, this tool is used to monitor and record the same system measurements that Performance Monitor collected for Windows NT 4.0 systems. As shown in Figure 11-3, System Monitor is a graphical tool that can monitor many different **events** concurrently. By using System Monitor, you can analyze network operations, identify trends and bottlenecks, determine system capacity, notify administrators when thresholds are exceeded, track the performance of individual system devices, and monitor either local or remote computers. To start System Monitor, first open Control Panel via the Start button by selecting Start, Settings, Control Panel. Next, open Administrative Tools by double-clicking its icon. Finally, double-click Performance to launch the Performance Console which contains System Monitor. (Hands-on Project 11-1 shows you how to use System Monitor to monitor memory performance.)

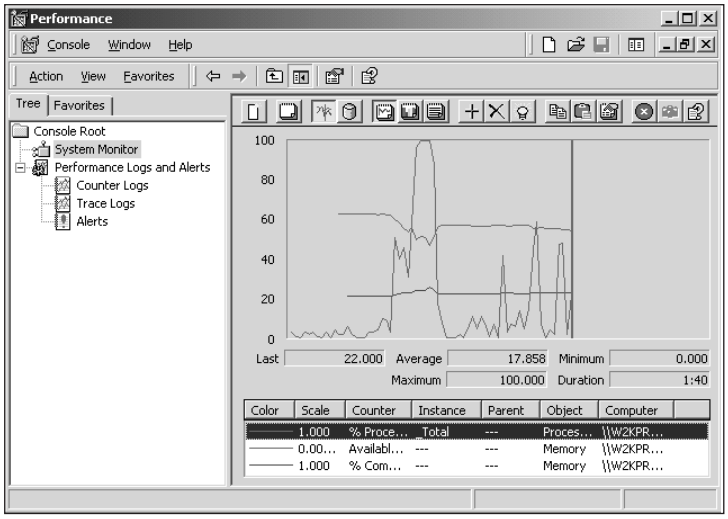

Figure 11-3 System Monitor

System Monitor is capable of a wide range of performance monitoring functions, including real-time monitoring, recording logs for future examination, and generating performance threshold alerts. Through proficient use of these functions, system administrators can effectively monitor their systems for bottlenecks, extract historical trending information, and be notified when abnormal activities occur, all of which are discussed in the following sections.

Realtime Monitoring

Realtime monitoring is the process of viewing the measured data from one or more counters in the System Monitor display area. System Monitor can display realtime (and logged) data in one of three formats: chart (see Figure 11-3), histogram (thermometer bars), or report (text-based instant values). You can select these views, or displays, by clicking the View Chart (default), View Histogram, or View Report buttons on the toolbar.

To begin monitoring a particular counter, click the Add Counters button that looks like a plus sign on the toolbar. You will see the Add Counters dialog box, shown in Figure 11-4. This dialog box reveals the object-oriented architecture of the Windows 2000 system as a whole and of performance monitoring in general. From this dialog box, you select counters based on the following:

- *Local or network accessible computer:* Counters can be read from the local system or any accessible system over a network.
- ***Object**:* An object is a component of the Windows 2000 system environment; objects range from devices to services to processes.
- ***Counter**:* Counters are aspects or activities of an object that can provide measurable information.
- ***Instance**:* An instance is a selection of a specific object when more than one is present on the monitored system; for example, multiple CPUs or hard drives.

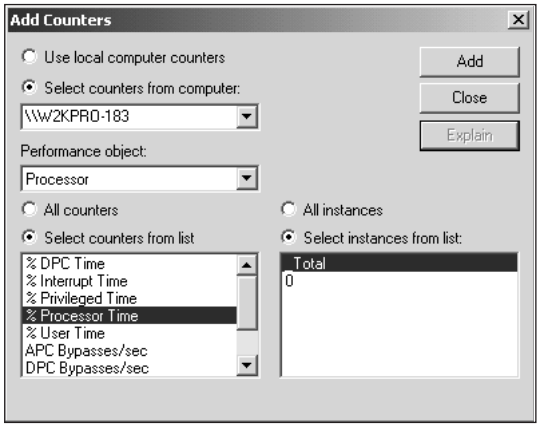

Figure 11-4 Add Counters dialog box

The Add Counters dialog box also allows you to select all counters for a specific object at once or all instances of an object at once. Once you've selected your host computer, object, counter (one or all), and instance (one or all), click the Add button to add the counter(s) and instance(s) to the list. If you need more information on a selected counter, click the Explain button. This reveals a floating window (see Figure 11-5) with additional information about the selected counter. Once you've added all the counters you are interested in monitoring, click Close to return to System Monitor.

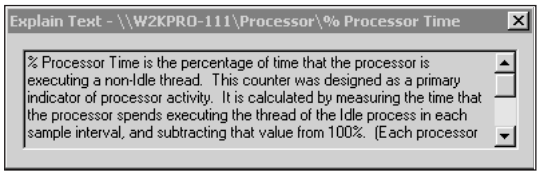

Figure 11-5 The Explain window

As you'll discover if you spend any time with System Monitor, its counters are legion—a plain-vanilla Windows 2000 installation makes it possible to monitor hundreds of such counters. In practice, however, there are only a handful of objects and associated counters that you work with regularly (some more regularly than others). The following list outlines the object and counter pairs that are worth memorizing, as well as several others that you may find useful when evaluating performance on your systems and networks. The list deals with six kinds of objects: LogicalDisk (the divisions of a drive into partitions or dynamic storage units), Memory (RAM), Network, PhysicalDisk (the actual hard disk as a whole), Processor (CPU), and System. For convenience, we present them in alphabetical order, listed in the form *Object: Counter*.

LogicalDisk and PhysicalDisk objects must first be enabled with the *diskperf* command, which is covered later in this chapter in the "Disk Bottlenecks" section.

- *LogicalDisk: Current Disk Queue Length*—If your system includes an older SCSI hard disk or any kind of EIDE or other controller type, measure this counter, which indicates how many system requests are waiting for disk access. If the queue length is greater than 2 for any logical drive, that drive is suffering from congestion. If you can't redistribute the load across multiple logical disks, consider upgrading your disk subsystem. Always check the corresponding PhysicalDisk counter when examining LogicalDisk counters.
- *LogicalDisk: % Disk Time*—This counter measures the percentage of time that a disk is busy handling read or write requests. It's unusual for this percentage to hit 100; it's also unusual for this level to be sustained at levels of 80% or higher. If this occurs, redistribute files to try to spread the load across multiple logical drives. Always check the corresponding PhysicalDisk counter.
- *LogicalDisk: Disk Bytes/Transfer*—This counter measures the average number of bytes transferred between memory and disk during read and write operations. If the value hovers at or near 4 KB (4086 bytes), this can indicate excessive paging activity on that drive. In general, a larger number indicates more efficient transfers than a smaller one, so look for declines against your baseline here.
- *Memory: Available Bytes*—This counter measures the number of bytes of memory available for use on the system at any given moment. Microsoft recommends that this value always be 4096 KB or higher. If values hover at or below this threshold,

that's a definite indicator that your system will benefit from additional RAM. You can obtain this number from the Task Manager Performance tab (it's the Available entry in the Physical Memory pane) without having to run System Monitor.

- *Memory: Cache Faults/sec*—This counter measures the number of times that the Windows 2000 cache manager must ask the system to bring a file's page in from disk or locate it elsewhere in memory. Higher values indicate potential performance problems, because a system's performance is best when cache hit rates are high. Establishing a baseline on a lightly loaded system will help you recognize when this counter begins to climb into risky regions (double, or more, the values that appear in the baseline). As with other memory counters, the proper response is to add more memory; in this case, adding more L2 cache is even better than adding main RAM.
- *Memory: Page Faults/sec*—This returns a count of the average number of page faults per second for the current processor instance. A page fault occurs whenever a memory page is referenced that is not already loaded in RAM. When this happens, the Virtual Memory Manager (VMM) must bring that page in from disk and possibly make room for that page by swapping an old page out to disk. This phenomenon helps to explain how memory congestion sometimes manifests itself through excessive disk activity. If this value increases to more than double what you observe in a light-load baseline, it can indicate a need for more RAM.
- *Memory: Pages/sec*—This tracks the number of pages that are written to or read from disk to satisfy requirements of the VMM, and also includes paging traffic for the system cache that occurs to access file data for applications. This is an important counter to watch if paging activity seems high, because it can indicate that paging levels are slowing the system down. If this value increases to more than double what you observe in a light-load baseline (or, in most instances, goes above 20 for a sustained period of time), it strongly indicates a need for additional RAM.
- *Network Interface: Bytes Total/sec*—This counts the total amount of traffic through the computer's network adapter, including all inbound and outbound data (framing characters as well as payload data). This measures the absolute amount of traffic moving through the adapter. When it begins to approach the practical maximum for the type of media in use, trouble lies ahead. It might require a switch to a faster type of network or indicate a need to distribute the machine's load across multiple network segments (and, therefore, multiple adapters).
- *Network Interface: Current Bandwidth*—This measures the current utilization levels of the network medium and provides a background count against which to evaluate the monitored machine's adapter. The same observations about loading and distribution apply to this counter as to the preceding one, except that this counter may indicate the need to partition the network to which this machine is attached, to lower the total traffic on individual cable segments.
- *Network Interface: Output Queue Length*—This measures the number of packets that are queued up for transmission across the network pending access to the medium. As with most other Windows 2000 queues, if this value approaches or exceeds 2, it

indicates that network delays are likely and that the bottleneck should be removed, if at all possible.

- *Network Interface: Packets/sec*—This measures the number of packets sent and received across a specific network adapter. Comparison with a baseline indicates when this value is getting out of hand. The observations that apply to the Bytes Total/sec counter apply to this counter as well.
- *PhysicalDisk: Current Disk Queue Length*—PhysicalDisk counters track hard disk activity on a per disk basis and provide much the same kind of information as the LogicalDisk counters. However, calculating pathological queue lengths for physical disks is different than for logical ones: here, the threshold for trouble is between 1.5 and 2 times the number of spindles on the hard drive. For ordinary drives, this is the same as for logical disks. But for RAID arrays (which Windows 2000 treats as a single drive), the number is equal to 1.5 to 2 times the number of drives in the array.
- *PhysicalDisk: % Disk Time*—This counter measures the percentage of time that a hard drive is kept busy handling read or write requests. For Windows 2000 machines, you may see peaks as high as 100%, but the sustained average should not exceed 80%. High sustained averages are not worrisome unless the corresponding queue length numbers are in the danger zone as well.
- *PhysicalDisk: Avg. # Disk Bytes/Transfer*—This counter measures the average number of bytes that read or write requests transfer between the drive and memory. This is a case where smaller values are more worrisome than larger ones, because they can indicate inefficient use of drives and drive space. If this behavior is motivated by applications, try increasing file sizes. If it's motivated by paging activity, an increase in RAM or cache memory is a good idea.
- *Processor: % Processor Time*—This counter measures the percentage of time that the CPU is busy handling nonidle threads—in other words, real work. Sustained values of 80% or higher indicate a heavily loaded machine. Consistent readings of 95% or higher indicate that a machine needs to have its load reduced or its capabilities increased (with a new machine, a motherboard upgrade, or a new CPU). See the section before the summary at the end of this chapter for a discussion of these various performance improvements.
- *Processor: Interrupts/sec*—This counter measures the average number of times per second that some device that requests immediate processing interrupts the CPU. Network traffic and system clock activity establish a kind of background count against which you should compare this number. Pathological increases occur when a malfunctioning device begins to generate false interrupts or when excessive network traffic overwhelms a network adapter. In both cases, this usually creates a count that's five or more times greater than a lightly loaded baseline situation.
- *System: Processor Queue Length*—This counter measures the number of execution threads that are waiting for access to a CPU. If this value increases to more than double the number of CPUs present on a machine (2 for a single-processor system), it indicates a need to distribute this machine's load across other machines, or

to increase its capabilities, usually by adding an additional CPU or by upgrading the machine or the motherboard. (Increasing CPU speed does not increase performance as much as you might think, because it does nothing for the machine's cache or its memory and bus transfer capabilities.)

Where we've indicated that more than one counter is worth watching for a particular object (for instance, there are four network-related counters), it's more significant when all counters experience a dramatic change in status simultaneously than when only one or two such counters show an increase. Across-the-board changes are more likely to indicate a bottleneck than are more localized ones (that are more likely to be caused by applications or by shifts in local conditions, traffic levels, and so forth).

You can customize the display of System Monitor through its Properties dialog box. Access the System Monitor Properties dialog box by selecting System Monitor in the left pane, then right-clicking in the right pane and selecting Properties from the resulting menu. The General tab (shown in Figure 11-6) offers the following controls:

- Set the view to Graph, Histogram, or Report (that is, the same function as the toolbar buttons).
- Enable the legend, value bar, and toolbar items.
- Set the report and histogram data to Default, Current, Average, Minimum, or Maximum.
- Set the appearance to 3D or Flat.
- Set the border to None or Fixed Single.
- Set the update/measurement interval in seconds; default is one second.
- Allow duplicate counter instances.

The Source tab (Figure 11-7) is used to set whether the displayed information is pulled from real-time measurements or pulled from a Counter log. (Counter logs are discussed in the "Logging and Using Logged Activity" section later in this chapter.) If a Counter log is used, you must also define the time range.

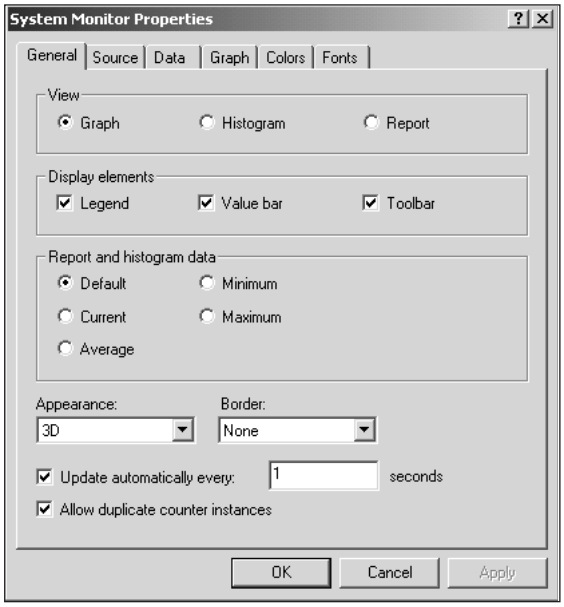

Figure 11-6 System Monitor Properties, General tab

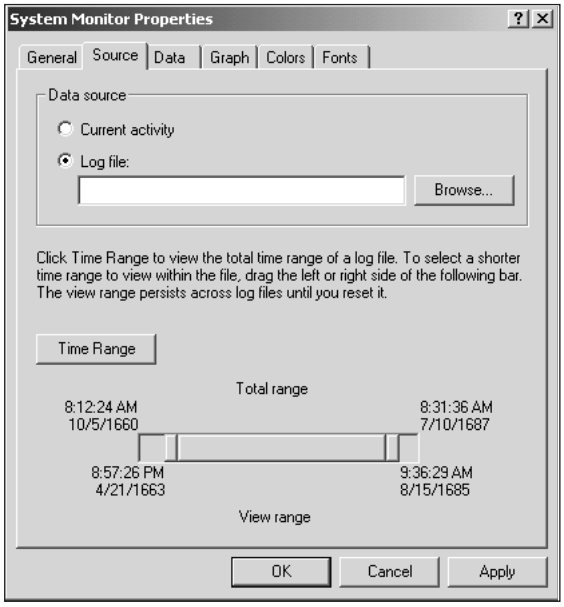

Figure 11-7 System Monitor Properties, Source tab

Use the Data tab to add or remove counters, as well as to alter the color, scale, width, and style (all via pull-down lists) of each counter's chart line. The Graph tab defines a title and vertical axis label, enables vertical and horizontal grid lines, indicates whether to display vertical scale

numbers, and sets the vertical maximum and minimum scale. Setting the vertical maximum and minimum scale is used to focus or expand the display for the purpose of making displayed counter measurements more informative. For example, if several counters display measurements within .3 deviation of the 80 mark, setting the maximum to 85 and the minimum to 75 expands the displayed information to grant an order of magnitude greater detail. The Colors tab defines the colors for the various components of System Monitor. The Fonts tab defines the font used to display text information. (Try Hands-on Project 11-2 to alter System Monitor display parameters.)

The System Monitor display in Chart (graph) view (refer to Figure 11-3) can show 100 data points from left to right. As each data point is measured, the event horizon line moves one point to the right as the data is added to the display. Below the graph of data in both Chart and Histogram views, five metadata items are listed. These are the value of the last, average, maximum, and minimum measurements of the selected counter and the total duration of the display field (calculated by multiplying the measurement interval by 100). Below these items is the counter legend, which lists all counters displayed in the graph, along with information about color, scale, counter name, instance, parent, object, and computer source. Selecting a counter in the legend causes the five metadata points to change their content.

Report view displays all selected counters grouped by instance, counter, object, and computer in text form. The information displayed in a report is the last measured value when viewing real-time data, or the averaged value over all data points in a time range when viewing logged data.

Logging and Using Logged Activity

The Windows 2000 Performance tool offers two types of logging capabilities. A **Counter log** records measurements on selected counters at regular, defined intervals. A **Trace log** records data only when certain events occur. Counter logs allow you to define exactly which counters are recorded (based on computer, object, counter, and instance). Trace logs record nonconfigurable data from a designated provider (such as the kernel) when an event occurs (such as process creation, thread deletion, disk I/O, and page fault). Trace logs are operating system environment status dumps that are more like a memory dump in the event of a Stop error than a log of performance statistics. You can review Counter log files using System Monitor, but Trace log files require a specialized tool to interpret the data. (Windows 2000 does not include a tool to read Trace log files.) Trace logs differ from counter data logs in that they measure data continually rather than take periodic samples. For more information on working with Trace logs, consult the *Windows 2000 Resource Kit*.

Using Counter logs is fairly simple. First, select the Counter Logs item beneath the Performance Logs and Alerts node of the Performance tool (see Figure 11-8). Notice that a Counter log named System Overview is already defined by default. You can use this predefined Counter log to get a basic look at the performance of the system. It's a basic look because it looks at only three counters—one for memory, one for storage, and one for CPU. Basically, the process requires selecting counters (based on computer, object, counter, and instance), setting the measurement interval, giving file storage information, and setting start and stop times.

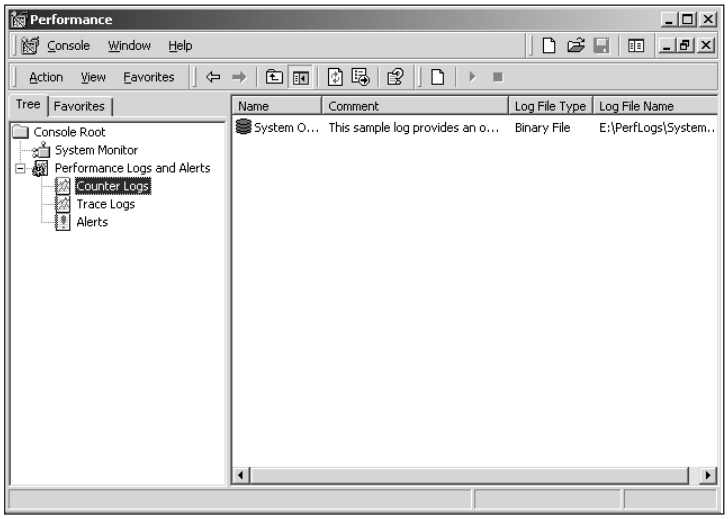

Figure 11-8 Counter Logs node of the Performance tool

Counter logs record data by taking measurements at regular intervals. The default interval is 15 seconds, but you can define intervals from 1 second to 999,999 days. All counters in a Counter log are measured at the same interval.

The Properties dialog box for Counter logs has three tabs. The General tab lists all counters included in the log, allows adding and removing counters, and sets the measurement interval. The Log Files tab defines the path and filename (with date stamp) of the log, sets the file type (comma-delimited, tab-delimited, or binary), and sets the maximum file size in KB or available drive space. The Schedule tab defines the start and the termination of a log (either manual or at a specified time). You can terminate a log manually, or set termination to occur after a specified length of time, at a specified time, or when the file is full. Once a log file closes, you can run a command (such as a batch file) or start a new log file (if drive space is available).

Once you define a Counter log, you can either wait for the defined start time or issue the Start command from the Action menu to begin recording data. Once recording, the Counter log will continue to collect data until either you manually stop the recording (by issuing the Stop command from the Action menu) or the defined stop event occurs. The Counter log will record data even when the Performance tool is closed. While a Counter log is recording data, the log icon beside the name will be green. When the recording stops, the icon is red. To learn how to create, start, and stop a Counter log, try Hands-on Project 11-3.

Once you've recorded a log file, it can be used in System Monitor. To do so, open Properties for System Monitor and go to the Source tab (refer to Figure 11-6). Select the Log file radio button, then provide the path to the Counter log file. Next, click the Time Range button to reveal the start and stop time stamps of the recorded data. Using the sliding endpoints, click and drag the view range. Time Range is used to focus the display around important data. Keep in mind that the display area can reveal only 100 measurement points. If you select

more than 100 data points, System Monitor will resample the data down to 100 points. For example, if you have 300 points, every three data items will be averaged to produce a single point. System Monitor retains all 300 data points, but only the averaged points are displayed. If fewer than 100 data points are selected in the time range, the data will be displayed without any extrapolation. (You can practice viewing data from a Counter log in System Monitor in Hands-on Project 11-4.)

Alerts

An **alert** is a watchdog that informs you when a counter crosses a defined threshold. Basically, an alert is an automated attendant looking for high or low values. An Alert object can consist of one or more counter/instance-based alert definitions. For example, you can configure an alert to be sent if the CPU goes above 90% usage, which is an indicator of CPU overload (see Figure 11-9). The individual alert definitions within an Alert object share the same sample interval, action triggers, and stop/start settings, but operate as distinct alert events. More than one Alert object can be created to assign different sample rates, action triggers, and stop/start settings.

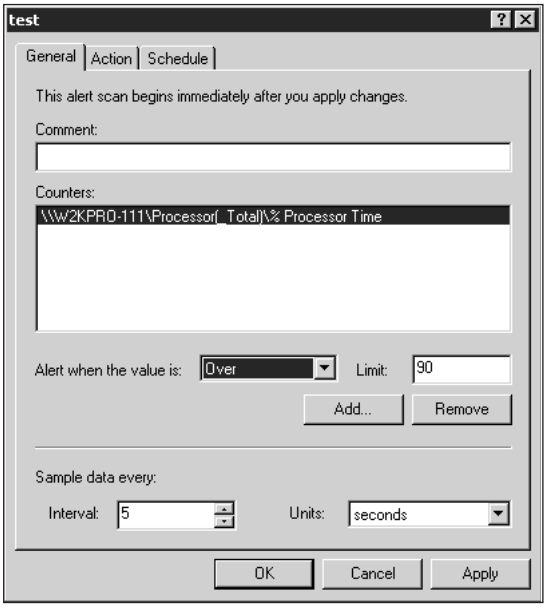

Figure 11-9 Setting an alert

An alert is defined on a counter/instance basis just like counter selection for the Counter log and System Monitor. An alert definition focuses on one or all counters of one or all objects on the local or networked computer. Each alert definition is assigned a threshold and told whether to issue an alert when the measured value is under or over that threshold. An alert event is triggered only when the measured value of the specific counter at the time of alert sampling has crossed the threshold. Counter levels between samplings have no effect on alerts because those

levels are unknown to the alert monitoring system. The sampling interval of an Alert object is the same as that of Counter Logs—one second to 999,999 days. (Try Hands-on Project 11-5 to create an Alert object.)

When an alert is triggered, any of four actions can occur. These are enabled and defined on the Action tab of an Alert object's Properties dialog box, as shown for the "test" Alert object in Figure 11-10.

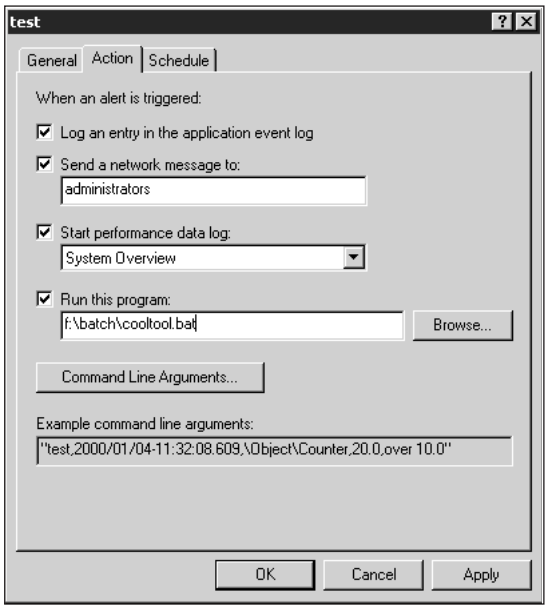

Figure 11-10 The Action tab of an Alert object's Properties dialog box

The possible actions of an alert are as follows:

- *Log an entry into the application event log:* You can view the event detail through Event Viewer.
- *Send a network message*: A single NetBIOS name of a user, group, or computer can be defined. When an alert occurs, a message regarding the alert and the measured counter level is sent.
- *Start performance data log:* Starts the recording of a Counter log.
- *Run this program:* Used to execute a program with command-line options or to launch a batch file. When this action is used, a string of performance-related information can be included at the end of the defined command line in the form. You can choose to have a single argument string with all data points separated with commas, or individual strings with the data elements of date/time, measured value, alert name, counter name, limit value, and a custom text string.

The Scheduling tab of an Alert event's Properties dialog box is much the same as that of a Counter log. Use this tab to define a start event that is either manual or at a specified time and a stop event that can be manual, after a length of time, or at a specified time. Similar to Counter logs, Alert events function even when the Performance tool is closed.

Event Viewer

The Windows 2000 **Event Viewer** is another useful tool for examining information about the performance and activities on a system. Event Viewer tracks all events generated by the operating system, as well as security and application events. An event is any activity that causes an event detail to be created in one of the logs of the Event Viewer. Failure of a device to load, an unsuccessful logon, or a corrupt database file can all be recorded by Event Viewer and viewed through one of three log files: System, Application, or Security. Access Event Viewer through the Administrative Tools via the Control Panel (try Hands-on Project 11-6). Figure 11-11 shows a typical Event Viewer displaying the System log.

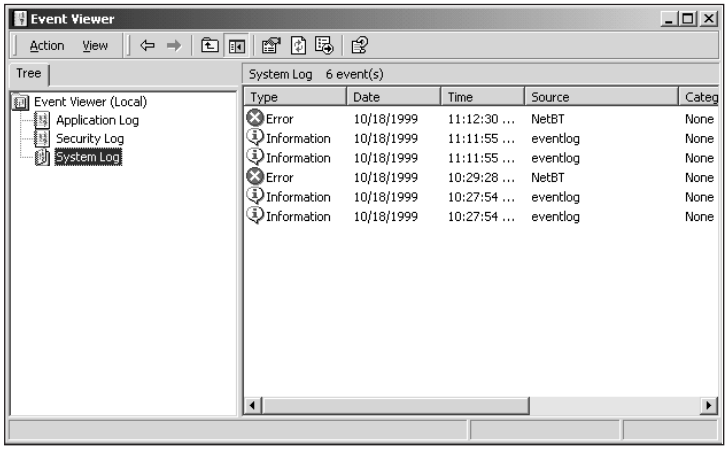

Figure 11-11 Event Viewer, System Log

There are three types of System and Application log events and two types of Security log events that are recorded in Event Viewer. These are listed in Table 11-1.

All Event log entries include the event's date and time, source, category (such as Logon or Logoff), an event number, the name of the account that generated the event, and the name of the computer on which the event occurred. You can use Event Viewer to view logs on other computers. To access log files on other computers, select the Connect to another computer command in the Action menu while Event Viewer (Local) is highlighted.

Each Event Viewer log has customizable properties. Access a log's Properties dialog box by highlighting that log then selecting the Properties command from the Action menu. The Properties dialog box (see Figure 11-12) has two tabs. Use the General tab to set properties such as the displayed name, the maximum file size, action to take when log is full (overwrite as needed, overwrite only events older than a specified number of days, or do not overwrite), and to manually clean out the log. Use the Filter tab to reduce the number of events displayed.

Filter options include sorting by the five event types, source of the event, event category, event ID, user, computer, and date range.

Table 11-1 Event Viewer Event Types

Event Type	Description
Information	Signifies rare but significant events about successful operation of internal services and drivers, indicated by the "i" icon. For example, when a database program loads successfully, it may generate an Information event.
Warning	Signifies potential problems although there is no present danger, indicated by an "!" (exclamation point) icon. For example, if disk space is running low, a Warning event may be logged.
Error	Signifies that significant problems exist that require immediate attention, indicated by a white "x" in a red circle. For example, if a driver fails to load correctly, an Error event is issued.
Success Audit	A Security log event that indicates that an event selected for audit has taken place. For example, when a user successfully logs on to a system, a Success Audit event is logged. A gold key icon represents success audits.
Failure Audit	A Security log event that indicates when an audited event has failed. For example, an unsuccessful attempt to access a network drive is logged as a Failure Audit event. A gold lock icon represents failure audits.

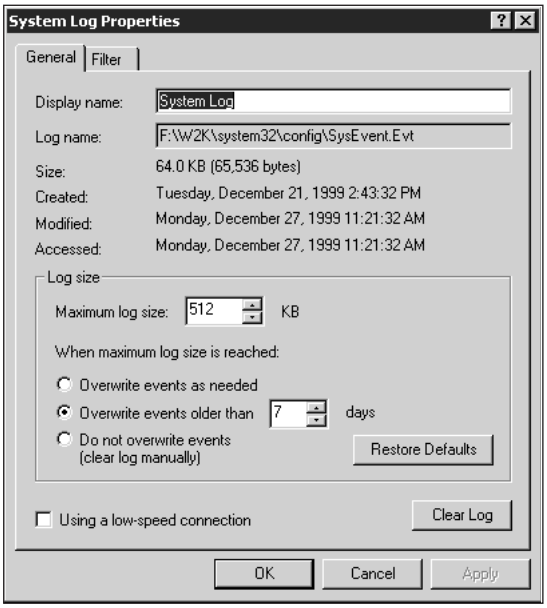

Figure 11-12 The Log Properties dialog box

The System log is the primary log file for most system services, drivers, and processes. Typical System log events occur when device drivers fail to load or load with errors, when system

services fail to start, when system service errors or failures occur, or when auditing is enabled and system-related events flagged for auditing occur. The Application log contains event messages that can be generated by Windows 2000 native applications or services. Unlike the System and Application logs, the Security log does not automatically track events. It records audit events such as logon, resource access, and computer restart and shutdown. Auditing must be enabled and configured (for details see Chapter 6).

Performance Options

You use the Performance Options dialog box (see Figure 11-13) to adjust system performance based on applications and virtual memory. Access this dialog box by clicking the Performance Options button on the Advanced tab of the System applet from the Control Panel. You can optimize general system performance by indicating whether the computer is used primarily for user-interactive applications or as a host for network services. The radio button selection of Applications grants foreground processes an additional priority boost, whereas Background services balances the use of processor resources. Use the Change button on the Performance Options dialog box to access the Virtual Memory dialog box, where the size and location(s) of the paging file is defined. See Chapter 3 for more details on optimizing the paging file size.

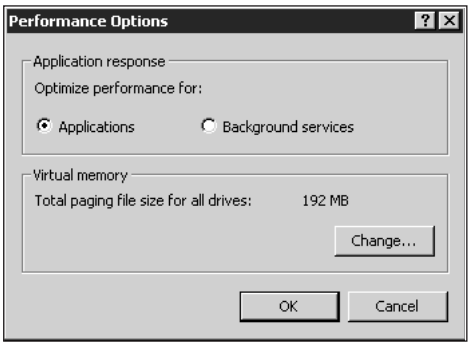

Figure 11-13 The Performance Options dialog box

Setting Application Priority

Windows 2000 uses 32 levels of application priority, numbered 0 to 31. These priority levels are used by Windows 2000 to determine which process should gain access to the CPU at any given moment. Users have only minimal control over the initial startup priority level of any launched task. The following list indicates important ranges and specific priority levels you should be aware of:

- 0–15: User-accessible process priorities
- 16–31: System-accessible process priorities
- 0–6: Low user range
- 4: Low value (as set in Task Manager, or with /low parameter to Start command)

- 5: BelowNormal value (as set in Task Manager)
- 7: Normal (default setting for user processes)
- 8–15: High user range
- 10: AboveNormal value (as set in Task Manager)
- 13: High value (as set in Task Manager, or with /high parameter to Start command)
- 16–24: Realtime values accessible to Administrator-level accounts
- 24: Realtime value (as set in Task Manager, or with /realtime parameter to Start command)
- 25-31: Realtime values accessible to operating system only

There are two techniques available to users and administrators to manipulate process priorities: You can manage already running processes using Task Manager, or use the Start command to launch processes with specific priority settings. One reason you may want to manipulate a process's priority is to give an application that is time-sensitive priority over another application that is not.

To use Task Manager, right-click any unoccupied region of the taskbar and select Task Manager from the menu. On the Processes tab of Task Manager, select the name of the desired process (usually this is the name of an .exe file that corresponds to the process), then right-click that process to produce another menu. From this menu, select the Set Priority item. This is where you can pick one of the predefined priority settings—Low, BelowNormal, Normal, AboveNormal, High, or Realtime. The current setting is the entry marked with a bullet symbol to the left. You must be logged on with Administrator privileges to use the Realtime setting.

You can use the Start command from a command prompt to launch a new application at some priority level other than the default. You can enter this command from either a command prompt or the Run command. The Start command follows this general syntax:

```
start /<priority-level> <program>
```

where /<*priority-level*> must be one of /low, /belownormal, /normal, /abovenormal, /high, or /realtime, and <*program*> is a valid path plus filename for the program you wish to launch at the specified priority level. For more details on the Start command, enter *start /?* from a command prompt.

Recognizing and Handling Bottlenecks

A **bottleneck** is a condition in which a limitation in a single component slows down an entire system. The first thing to remember about bottlenecks is that they will always exist in any computer. Applications, hard drives, operating systems, and network interfaces might all act as bottlenecks from time to time, but for any given configuration, it is always possible to identify one component that slows the others down.

There is no single bottleneck monitor that can easily identify all possible problems. However, by using the monitoring tools included with Windows 2000, you can identify possible bottlenecks and make adjustments. The goal in performance tuning a workstation is to make bottlenecks unnoticeable for everyday functions. A computer used for CAD requires much greater throughput than a computer used primarily for word processing. Ideally, a computer should be waiting for user input, rather than making users wait for the computer's response. In this ideal case, the user becomes the bottleneck (and because computers cannot do much to speed up humans, this is regarded as an ideal situation).

Although the details will vary from situation to situation, the process of finding and fixing computer system bottlenecks follows a reasonably consistent course, and usually works something like this:

1. Create a baseline for a computer. For Windows 2000, this includes observations of memory usage, disk usage, CPU usage, operating system resource usage and activity, and network utilization, at the barest minimum.
2. The first step in determining potential bottlenecks is to compare baseline observations to current system behavior. In most cases, one or more of the baseline values will have changed for the worse. These changes indicate further areas of investigation.
3. Investigate the more common causes of system problems (some of these for Windows 2000 are documented later in this chapter) to see if any match the symptoms your computer is exhibiting. If you have a match, the causes of bottlenecks are easy to identify and fixes are easy to apply.
4. If the list of "usual suspects" does not produce an obvious culprit, further analysis is required. Part of using System Monitor and other performance tools is knowing how to use them to obtain more detailed statistics on system behavior and being able to analyze their reports and statistics to pinpoint potential bottlenecks. Use the general analytical techniques and combinations of objects and counters described in this chapter to help in isolating and identifying bottlenecks.
5. Once a potential bottleneck is identified, you make changes to the system configuration to correct the situation. Sometimes this involves software configuration changes; other times it can involve adding or replacing specific hardware components or subsystems.
6. Always test the impact of any fix you try on the system. Compile a new set of statistics and compare them to the same system measurements before the fix was applied. Sometimes, the fix does the trick and values return to normal, or at least come closer to acceptable levels. At other times, the fix doesn't make a difference. In that case, further analysis, other fixes, and more testing are required. It's important to keep at the job until something improves the bottleneck conditions.

It's important to understand that bottlenecks can always be fixed, but some fixes are more expensive than others. Remember, you can always replace an overloaded server or workstation with another, bigger, faster system, or you can spread the load of a single overloaded system across multiple systems to reduce the impact on any single machine. These kinds of fixes are a

great deal more expensive than tweaking system settings or adding more memory or disk space to a machine. However, in some cases, drastic solutions are necessary. If you do your job of monitoring performance correctly, such radical changes needn't take anyone by surprise.

Common Bottlenecks

In this section, we explain how to use the counters you have chosen to watch, either alone or in combination, to determine what kinds of bottlenecks might be present on a system. We also discuss steps you might consider taking to correct such bottlenecks. We tackle these subjects in the following order:

- Disk bottlenecks
- Memory bottlenecks
- Processor bottlenecks
- Network bottlenecks

Disk Bottlenecks

Disk bottlenecks are the most likely problem when disk-related counters increase more dramatically than other counters, compared to your baseline, or when disk queue lengths become unacceptably long. Windows 2000 collects information about the performance of physical disks (the actual devices) by default. However, to view information about logical disks, you must enable the LogicalDisk object with the *diskperf* command. When you use the *diskperf* command, you must reboot the system for any changes to take effect. The syntax of the *diskperf* command is:

```
diskperf [-y[d|v]] | [-n[d|v]] [\\computername]
```

where:

- -Y enables both PhysicalDisk and LogicalDisk objects.
- -YD enables only the PhysicalDisk object.
- -YV enables only the LogicalDisk object.
- -N disables both the PhysicalDisk and LogicalDisk objects.
- -ND disables only the PhysicalDisk object.
- -NV disables only the LogicalDisk object.
- *computername* performs the object enable/disable on the specified system; this applies to the local system if this item is not included.

You can execute the *diskperf* command from a command prompt or from the Start, Run command. Windows 2000 does not enable the LogicalDisk object by default because measuring that object causes a measurable degradation in storage device performance. You should disable the LogicalDisk object once you have completed your monitoring.

If Disk Queue Length and % Disk time values remain consistently high (1.5 or higher and more than 80%, respectively), it's probably time to think about adding more disk controllers or drives or possibly switching existing drives and controllers for newer, faster SCSI equivalents. This costs money, but can provide dramatic performance improvements on systems with disk bottlenecks. Adding a controller for each drive can substantially improve performance, and switching from individual drives to disk (RAID) arrays can also improve performance on such systems. Because high-end disk controllers often include onboard memory that functions as yet another level of system cache, they can confer measurable performance benefits.

Software can also contribute to disk bottlenecks, often because of poor design, configuration settings that affect disk performance, or outdated drivers. Because tweaking an application's source code is beyond the reach of most system administrators, inspect the application to see if you can increase the size of the files it manipulates directly or the size of data transfers it requests.

Memory Bottlenecks

Windows 2000 is subject to several different kinds of **memory bottlenecks**. To begin with, it's important to make sure that the paging file is working as efficiently as possible; that is, its size is two to three times the amount of physical RAM on a machine (see Chapter 3). On machines with more than one drive, Microsoft recommends not situating the paging file on the drive where the Windows 2000 system files reside. If multiple drives are available, it's a good idea to spread the paging file evenly across all such drives (except the drive with the systems files). Better yet is for each drive to have its own disk controller; this allows Windows 2000 to access all drives in parallel.

You can detect excessive paging activity by watching the page-related counters mentioned earlier and by observing the lowest number of Available Bytes over time. (Microsoft recommends that this number never dip below 4 MB or 4096 KB.) Excessive disk time and disk queue lengths can often mask paging problems, so be sure to check paging-related statistics when disk utilization zooms. Adding more memory can fix such problems and improve overall system performance.

Processor Bottlenecks

Processor bottlenecks are indicated when the Processor object's % Processor time counter stays consistently above 80% or when the System object's Processor Queue Length counter remains fixed near a value of 2 or more. In both cases, the CPU is being overworked. However, occasional peaks of 100% for processor time are not unusual (especially when processes are being launched or terminated). The combination of high utilization and overlong queues is more often an indication of trouble than is an occasionally high utilization rate.

Even on machines that support multiple CPUs, it's important to recognize that performance doesn't scale arithmetically as additional CPUs are added. A second CPU gives a more dramatic incremental improvement in performance than a third or fourth; however, two CPUs do not double performance (nor do three, for that matter). You're often better off responding

to CPU bottlenecks by redistributing a machine's processing load or by replacing the machine or upgrading its CPU, memory, and motherboard. Simply upgrading or adding another CPU neither increases the amount of cache memory on a system nor improves the system's underlying CPU-to-memory data transfer capabilities, both of which often play a crucial role in improving system performance.

When more than one CPU is present on a system, you can choose to monitor the activity of the CPUs on either an individual basis or as a whole group. To monitor a single CPU, select the individual instance of the CPU. The first CPU is instance 0; the second CPU is instance 1. To monitor the activity of all CPUs as a whole, select the _Total instance.

Network Bottlenecks

Network bottlenecks are not typical on most Windows 2000 machines, because end users seldom load the network sufficiently to experience performance problems. However, it is worth monitoring how much traffic is passing through a workstation's network adapter as compared to the networking medium to which it is attached. Excessive activity can indicate a failing adapter (sometimes called a "jabbering transceiver") or an ill-behaved application. In both cases, the fix is relatively straightforward—replace the NIC or the application, respectively. Occasionally, however, the network itself may be overloaded. This situation is indicated by utilization rates that exceed the recommended maximum for the medium in use. (For example, Ethernet should not be loaded more heavily than 56% utilization; token ring can function adequately at loads as high as 98% utilization.) When this happens, as a network administrator you have two options: divide the network into segments and balance traffic so that no segment is overloaded, or replace the existing network with a faster alternative. Neither of these options is especially fast, cheap, or easy, but the former is cheaper than the latter, and may give your network—and your budget—some breathing room before a wholesale upgrade is warranted.

Eight Ways to Improve Windows 2000 Performance

Although there are many things you can do to deal with specific system bottlenecks, there are eight particularly useful changes in system components, elements, approaches, or configuration that are likely to result in improved performance. These are listed in approximate order of their potential value, so always try to hit elements higher in the list first when you are attempting to boost Windows 2000 performance. All elements on this list are worth considering when performance improvements are needed.

- *Buy a faster machine:* It takes only a year or so for a top-of-the-line, heavily loaded PC to become obsolete these days. When you find yourself considering a hardware upgrade to boost performance, compare the price of your planned upgrade to the cost of a new machine. If you're planning on spending more than half the cost of a newer computer (and can afford to double your expenditure), buy the newer, faster machine. Otherwise, you may be facing the same situation again in a few months. The extra cost buys you at least another year before you must go through this exercise again.

- *Upgrade an existing machine:* You might decide to keep a PC's case, power supply, and some of the adapter cards it contains. As long as the price stays below half the cost of a new machine, replacing a PC's motherboard not only gets you a faster CPU, more memory capacity (both cache and main memory), but it can also get you more and faster bus slots for adapter cards. While you're at it, be sure to evaluate the costs of upgrading the disk controller and hard drives, especially if they're more than twice as slow as prevailing access times. (As we write, garden-variety drives offer average access times of around 10 milliseconds, and fast drives offer average access rates of 3 to 4 milliseconds.)
- *Install a faster CPU:* As long as you can at least double the clock speed of your current CPU with a replacement unit, such an upgrade can improve performance for only a modest outlay. Be sure to review your memory configuration (cache and main memory) and your disk drives at the same time. A faster CPU on an otherwise unchanged system can't deliver the same performance boost as a faster CPU with additional memory and faster drives.
- *Add more L2 cache:* Many experts believe that the single most dramatic improvement for an existing Windows 2000 PC comes from adding more L2 cache to a machine (or to buy only machines with the maximum amount of L2 cache installed). The CPU can access L2 cache in two CPU cycles, whereas access to main RAM usually takes 8 to 10 CPU cycles. This accounts for why adding L2 cache to a machine can produce dramatic performance improvements. Although cache chips are quite expensive, they provide the biggest potential boost to a system's performance, short of the more drastic—and expensive—suggestions detailed earlier in this list.
- *Add more RAM:* Windows 2000 is smart about how it uses main memory on a PC. It can handle large amounts of RAM effectively, and it has been widely observed that the more processes that are active on a machine, the more positive the impact of a RAM increase. For moderately loaded workstations (six or fewer applications active at once), 64 MB of RAM is recommended. For heavily loaded workstations, 128 MB or more may improve performance significantly.

When you add RAM to a Windows 2000 machine, make sure to resize the paging file to properly accommodate the change in physical RAM.

- *Replace the disk subsystem:* Because memory access occurs at nanosecond speeds, and disk access occurs at millisecond speeds, disk subsystem speeds can make a major impact on Windows 2000 performance. This is particularly true in cases where applications or services make frequent accesses to disk, when manipulating large files, or when large amounts of paging activity occur. Because the controller and the drives both influence disk subsystem speeds, we recommend using only Fast Wide SCSI drives and controllers (or the latest of the EIDE drives and controllers) on Windows 2000 machines. However, it's important to recognize that a

slow disk controller can limit a fast drive and vice versa. That's why upgrading the entire subsystem is often necessary to realize any measurable performance gains.

- *Increase paging file size:* Whenever System Monitor indicates that more than 10% of disk subsystem activity is related to paging, check the relationship between the Limit and Peak values in the Commit Charge pane in Task Manager. (Right-click on any empty portion of the taskbar, select Task Manager, then select the Performance tab and check the lower-left corner of the display.) If the Peak is coming any closer than 4096 KB to the limit, it's time to increase the size of this file. We recommend using a figure somewhere between twice and three times the amount of RAM installed in the machine.
- *Increase application priority:* On machines where a lot of background tasks must be active, you can use the Task Manager's Processes tab to increase the priority of any already running process. Highlight the process entry, then right-click to produce a menu that includes a Set Priority entry. This entry permits you to set the priority to High or Realtime, either of which can improve a foreground application's performance. We recommend that you set only critical applications to Realtime, because they can interfere with the operating system's ability to do its job. To launch an application with an altered priority level, refer to the section, "Setting Application Priority," earlier in this chapter.

Only users with administrator level access to Windows 2000 can run processes at a Realtime priority level. Be aware that raising the priority of a single process causes other background processes to run more slowly. The other performance improvements in this list should improve system performance across the board; this one is limited to those processes whose priorities are increased.

Chapter Summary

- Windows 2000 Professional provides a number of tools to monitor system performance. By using these tools, it is easy to alleviate the effects of bottlenecks and to improve system response time.
- You can use Task Manager to view applications, processes, and overall system performance, or to stop applications and processes, an efficient way to regain control from an application that is experiencing problems. The default configuration of the Processes tab not only displays the names of running processes, but also their process IDs, percentages of CPU and CPU time used, and memory consumption. Other columns, such as Virtual Memory Size and Thread count, can be added to the Processes tab.
- The Performance console is an exceptionally useful collection of tools that include System Monitor, log files, and alerts. System Monitor is used to watch real-time performance or review data collected in log files. Log files record performance data for one or more counters over a specified period of time. Alerts inform administrators when specific counters cross defined threshold levels.

- The Event Viewer is a less dynamic, but equally important tool that tracks logs generated by the system. Event Viewer monitors three different logs: System, Application, and Security. The System log records system information and errors, such as the failure of device driver to load. The Application log maintains similar information for programs, such as database applications. The Security log monitors system security events and audit activities.
- Finally, you should keep an eye on logs and performance counters to isolate any bottlenecks that occur in the system. Once isolated, take the steps necessary to remove the bottleneck and get the system running more smoothly. In addition, try the recommendations listed in this chapter for improving overall system performance.

Key Terms

alert — A watchdog that informs you when a counter crosses a defined threshold. An alert is an automated attendant looking for high or low values, and can consist of one or more counter/instance-based alert definitions.

baseline — A definition of what a normal load looks like on a computer system; it provides a point of comparison against which you can measure future system behavior.

bottleneck — A system resource or device that limits a system's performance. Ideally, the user should be the bottleneck on a system, not any hardware or software component.

counter (or **performance counter)** — A named aspect or activity that the Performance tool uses to measure or monitor some aspect of a registered system or application object.

Counter log — A log that records measurements on selected counters at regular, defined intervals. Counter logs allow you to define exactly which counters are recorded (based on computer, object, counter, and instance).

disk bottleneck — A system bottleneck caused by a limitation in a computer's disk subsystem, such as a slow drive or controller, or a heavier load than the system can handle.

event — A system occurrence that is logged to a file.

Event Viewer — A system utility that displays one of three event logs: System, Security, and Application, wherein logged or audited events appear. The Event Viewer is often the first stop when monitoring a system's performance or seeking evidence of problems because it is where all unusual or extraordinary system activities and events are recorded.

handle — A programming term that indicates an internal identifier for some kind of system resource, object, or other component that must be accessed by name (or through a pointer). In Task Manager, the number of handles appears on the Performance tab in the Totals pane. A sudden increase in the number of handles, threads, or processes can indicate that an ill-behaved application is running on a system.

instance — A selection of a specific object when more than one is present on the monitored system; for example, multiple CPUs or hard drives.

memory bottleneck — A system bottleneck caused by a lack of available physical or virtual memory that results in system slowdown or (in extreme cases) an outright system crash.

network bottleneck — A system bottleneck cause by excessive traffic on the network medium to which a computer is attached, or when the computer itself generates excessive amounts of such traffic.

object — A component of the Windows 2000 system environment; objects range from devices to services to processes.

process — An environment that defines the resources available to threads, the executable parts of an application. Processes define memory available, show where the process page directory is stored in physical memory, and other information that the CPU needs to work with a thread. Each process includes its own complete, private 2 GB address space and related virtual memory allocations.

processor bottleneck — A system bottleneck that occurs when demands for CPU cycles from currently active processes and the operating system cannot be met, usually indicated by high utilization levels or processor queue lengths greater than or equal to two.

System Monitor — The utility that tracks registered system or application objects, where each such object has one or more counters that can be tracked for information about system behavior.

thread — In the Windows 2000 run-time environment, a thread is the minimum unit of system execution and corresponds roughly to a task within an application, the Windows 2000 kernel, or within some other major system component. Any task that can execute in the background can be considered a thread (for example, run-time spell checking or grammar checking in newer versions of MS Word), but it's important to recognize that applications must be written to take advantage of threading (just as the operating system itself is).

Trace log — A log that records data when only certain events occur. Trace logs record nonconfigurable data from a designated provider when an event occurs.

Review Questions

1. Monitoring is the act of changing a system's configuration systematically, and carefully observing performance before and after such changes. True or False?
2. In a system that is performing optimally, the user should be the bottleneck. True or False?
3. Which of the following can Task Manager monitor?
   a. application CPU percentage
   b. total CPU percentage
   c. process CPU percentage
   d. all of the above
4. The longer a system is in productive use, the more its performance _____.

5. Which of the following are methods to access Task Manager?
   a. Ctrl+Alt+Delete
   b. executing "taskman" from the command prompt
   c. Ctrl+Shift+Esc
   d. Control Panel
6. In System Monitor, the counters are the same for all objects. True or False?
7. A(n) _____ event is issued when a driver fails to load.
8. The _____ provides a detailed description of a counter.
9. To record log files the Performance tool must be open. True or False?
10. A Counter log can include which of the following?
    a. one or more counters
    b. counters from multiple computers
    c. different intervals for each counter
    d. a stop time defined by a length of time
11. A _____ occurs when a system resource limits performance.
12. Which of the following objects cannot be used to obtain measurements of performance by default?
    a. Memory
    b. LogicalDisk
    c. RAS port
    d. System
13. In general, a bottleneck might exist if a queue counter is consistently _____ than the total number of instances of that object.
14. Which one of the following counters is the most likely indicator of a high level of disk activity caused by too little RAM?
    a. Memory: Pages/sec
    b. Memory: Page Faults/sec
    c. Memory: Cache Faults
    d. Memory: Available bytes
15. Which of the following tools can monitor another computer's information?
    a. System Monitor
    b. Task Manager
    c. Event Viewer
16. The _____ on the Source tab is used to select a window of data from a Counter log.
17. The _____ is used to generate system performance reports.

18. What parameter should be used with diskperf to disable only the PhysicalDisk object?
    a. -yd
    b. -yv
    c. -nd
    d. -nv
19. The System Monitor can display only _____ data points.
20. The _____ and _____ event types are available only in the Security log.
21. Of the following commands, which gives the Test.exe application the highest priority level available to ordinary users (not administrators)?
    a. start /abovenormal test.exe
    b. start /normal test.exe
    c. start /high test.exe
    d. start /realtime test.exe
22. Which of the following activities can occur when an alert is triggered?
    a. an alert to a NetBIOS name
    b. shutdown of the system
    c. start the recording of a Counter log
    d. write an event to the Application log
23. The _____ feature of Event Viewer can be used to quickly locate all audit details for a specific user.
24. The Start command can be used to alter the priority of active processes. True or False?
25. What change to a system is most effective in producing a performance improvement?
    a. adding RAM
    b. replacing network cables
    c. adding more processors
    d. updating drivers

HANDS-ON PROJECTS

Project 11-1

To use System Monitor to monitor performance of memory, processor, disks, network, and applications:

1. Open the Control Panel by selecting **Start**, **Settings**, **Control Panel**.
2. Double-click the **Administrative Tools** icon.
3. Double-click the **Performance** icon.

4. Select the **System Monitor** node in the MMC console.
5. Click **Add** on the toolbar (it's the plus sign).
6. Select the **% Processor Time** counter from the **Processor** object, which is selected by default.
7. Use the **Performance** object pull-down list to select the **Memory** object.
8. Select the **Pages/sec** counter.
9. Click **Add**.
10. Click **Explain**. Read the detail about the selected counter.
11. Repeat Steps 7 through 10 to add some or all of the following counters (*Note*: if multiple instances of these objects are present, select one or more instances and/or the _Total instance):

    PhysicalDisk: Current Disk Queue Length
    PhysicalDisk: %Disk Time
    PhysicalDisk: Avg. Disk Bytes/Transfer
    Memory: Available Bytes
    Memory: Cache Faults/sec
    Memory: Page Faults/sec
    Memory: Pages/sec
    Network Interface: Bytes Total/sec
    Network Interface: Current Bandwidth
    Network Interface: Output Queue Length
    Network Interface: Packets/sec
    Processor: Interrupts/sec
    System: Processor Queue Length
    Thread: % Processor Time
    Thread: Priority Current
    Process: % Processor Time
    Process: Elapsed Time
    Process: Page Faults/sec
    Process: Thread Count
12. Click **Close**.
13. Launch and close **Windows Explorer** or any other application several times, read files from disk, access network resources, and so on to cause system activity.
14. Notice how the respective lines of the selected counters change according to system activity.

Project 11-2

To use System Monitor to alter the display parameters:

1. Click the **Properties** button on the toolbar.
2. Change update automatically from every 1 second to **2** seconds.
3. Select the **Data** tab.

4. Select the **Memory: Pages/sec** counter.
5. Change the color, width, and style, using the pull-down lists.
6. Select the **Graph** tab.
7. Select the **Vertical** grid and **Horizontal** grid check boxes.
8. Click **OK** to close the Properties dialog box.

Project 11-3

To create, start, and stop a Counter log:

1. Launch the Performance tool if it is not still open from the previous Hands-on Project.
2. Click the boxed plus sign next to the Performance Logs and Alerts node to expand its contents.
3. Select the **Counter Logs** item.
4. Select **New Log Settings** from the Action menu.
5. Type a name, such as **Set1**. Click **OK**.
6. Click the **Add** button on the General tab.
7. Click the **Add** button on the Select Counters dialog box to add the % Processor Time counter, which is selected by default, to the log.
8. Click **Close**.
9. Change the interval from 15 seconds to **2** seconds.
10. Select the **Log Files** tab. Review its controls, but don't make any changes.
11. Select the **Schedule** tab.
12. If you are prompted that the log file path does not exist but can be created, select **Yes** to create the path.
13. In the Start log area, select the **At** option and change the start time to **3** minutes from the present.
14. In the Stop log area, select the **After** option and change the time to **4** minutes.
15. Click **OK**.
16. Notice the new log appears in the list. Within three minutes, its icon will turn green.
17. After the icon turns green, launch and terminate Windows Explorer several times to cause system activity.
18. After four minutes the icon turns back to red. Do not go on with the next Hands-on Project until the icon is red again.

Project 11-4

To view data from a Counter log with System Monitor:

1. Launch the Performance tool if it is not still open from the first Hands-on Project.
2. Select the **System Monitor** node.

3. Right-click the right pane and select **Properties** from the resulting menu.
4. Select the **Source** tab.
5. Select the **Log file** option.
6. Use the **Browse** button to locate and select the Counter log created in Hands-on Project 11-3. Click **Open**.
7. Click **OK** in the System Monitor Properties dialog box.
8. Click the **New Counter Set** button in the toolbar (the blank page with a sparkle on the top-right corner).
9. Click the **Add** button (the plus sign) on the toolbar.
10. Click **Add** to add the % Processor Counter to the System Monitor display. Note the Counter log recorded in the previous Hands-on Project has only this one counter so it is selected by default.
11. Click **Close**.
12. Because the Counter log recorded measurements every 2 seconds for 4 minutes, there are 120 data points that are compressed and averaged to create the display you see. To prevent compression of data, you must select a time range of 100 data points or fewer.
13. Click the **Properties** button on the toolbar.
14. Select the **Source** tab.
15. Click the **Time Range** button to refresh the Counter log data.
16. Click and drag the right slider so that only 198 seconds separate the start and stop ends of the view range.
17. Click **OK**.
18. Notice that now 99 data points are displayed.

Project 11-5

To create an Alert object:

1. Launch the Performance tool if it is not still open.
2. Select the **Alerts** node.
3. Select **New Alert Settings** from the Action menu.
4. Type a name such as **Set1**. Click **OK**.
5. Click **Add**.
6. Click **Add** to add the % Processor Time counter to the alert. Note that this counter is selected by default.
7. Click **Close**.
8. Select **Over** in the "Alert when the value is" pull-down box.
9. Type in **50** in the Limit box.
10. Change the update interval to **1** second.
11. Select the **Action** tab.

12. Select the **Send a network message to** check box.
13. Type in the **username** of the account with which you are currently logged on.
14. Select the **Schedule** tab.
15. Select the **Manually (using the shortcut menu)** option in the Start scan area.
16. Click **OK**.
17. Select the new **Alert object** that appears in the list of alerts.
18. Select the **Start** command from the Action menu. Its icon will be green when active.
19. Launch and terminate Windows Explorer several times to force system activity. When the % Processor Usage crosses the 50 percent threshold, a network message will appear on your screen. Click **OK** to close it.
20. Select the **Delete** command from the Action menu. Click **OK** to confirm the deletion. This deletes the Action object.

Project 11-6

To use Event Viewer to view an event detail:

1. Open the **Control Panel** by clicking the **Start** button, clicking on **Settings**, and then clicking on **Control Panel**.
2. Open the **Administrative Tools** by double-clicking its icon in the Control Panel.
3. Open **Event Viewer** by double-clicking on its icon in the Administrative Tools window.
4. Select the **Application log**.
5. Locate and select an Information detail with a SysmonLog source.
6. Double-click the item to open the event detail.
7. Notice that the Description includes information about the counter and the measured level that caused the alert.
8. Click **OK**.
9. Close the Event Viewer.

CASE PROJECTS

1. Performance on a Windows 2000 system used by the accounting department has been slowly degrading. You recently added a 100-Mbps network card, thinking that would correct the problem. To your knowledge, no other hardware has been added to the server, but you suspect someone has been adding software.

   Describe the steps you will use to determine what is causing the system to slow down, including which monitoring applications you will use and on which computer they will be run.
2. You are considering upgrading your Windows 2000 hardware, including memory, hard drive controller, and video card. The only things you are planning to keep are your hard drive, motherboard, and CPU.

   Outline the tools and utilities you will use to measure the performance increase or decrease, as each new component is added. Include information on expected performance changes and actual changes.